REMARKS

Applicant thanks the examiner for the examiner's time and helpful comments during the November 16, 2006, November 17, 2006, and January 9, 2007, telephone interviews to discuss the claims and the October 13, 2006, Office Action. Applicant's attorneys, Denis G. Maloney and Ido Rabinovitch, and the examiner discussed amendments to the independent claims to put the application in condition for allowance. No immediate agreement was reached.

Claims 1, 4-8, 10-16, 18-22 and 24 are pending in the above-referenced patent application. Claims 1, 15, 22 and 24 are independent. Claims 4-7, 10, 12-14, 18, 19 and 21 have been deemed allowable.

The examiner rejected claim 24 under 35 U.S.C. §101 on the ground that the claimed invention is directed to non-statutory subject matter.

Specifically, the examiner states:

> 3. As to claim 24, the claimed computer product residing on a computer readable medium for causing a multithreaded parallel processor to perform a function is not necessarily restricted a hardware . For example, page 4, lines 10-19 recited the application or the function was for the hardware based multithreaded processor. However, page 4, lines 22-25 recites another example of application was directed to a matching engine for electronic trading . The electronic trading is not tangible, and no clear definition ahs been set forth in the specification as what readable medium is. Therefore, it is not sure what embodiment the computer readable medium is directed to . If the computer readable medium is directed to the electronic trading , it is not tangible. Furthermore, although claim recites the computer readable medium was for causing the multithreaded parallel processor to perform a function, and to cause the processor to receive, perform, and wake up, it is an intended result, not a positive recitation of the limitation. The focus is not on the steps or features taken to achieve the final result which is useful, concrete and tangible, but rather that the final result achieved which is useful, concrete and tangible (see page 20, 101 Interim Guidelines). (Office Action, pages 2-3)

Applicant amended independent claim 24 to replace the wording "medium" with "storage device." Applicant's independent claim 24 thus recites a "computer readable storage device" which is a tangible element. Additionally, the instructions stored on the medium are such that they cause a multi-threaded processor, which is a tangible device, to perform certain operations. Particularly, the instructions cause the processor to, among other things perform a swap operation that results in the context of a first thread being swapped out, and the context of a

second thread swapped in. The second thread then begins execution on the processor. These operations are all concrete, useful and tangible actions, which is all that is necessary to make a claim statutory. Therefore, applicant's independent claim 24 recites statutory subject matter.

The examiner rejected claims 1, 8, 15, 16, 20, 22 and 24 under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 4,868,735 to Moller, in view of U.S. Patent No. 5,872,963 to Bitar, in view of U.S. Patent No. 5,247,671 to Adkins and further in view of U.S. Patent No. 6,005,575 to Colleran. Additionally, the examiner rejected claim 11 under 35 U.S.C. §103(a) as being unpatentable over Moller, in view of Bitar, in view of Adkins, in view of Colleran and further in view of U.S. Patent No. 6,505,229 to Turner.

Applicant amended independent claim 1 to remove the recitation "voluntary swap" and to clarify that the parameter specified in the context-swap instruction is a "user-specified" parameter. Support for this clarification is provided in page 18, line 31, to page 19, line 29 of the application, which discusses the context swap instruction "CTX_ARB" and the various parameters that can be specified by the user. Applicant similarly amended independent claims 15, 22 and 24. In addition, claims 4-8, 10-13 and 18-20 have been amended to remove the recitation "voluntary swap" and to add the recitation "user-specified" before the word "parameter" so as to make the language recited in those claims consistent with the language recited in the respective amended independent claims. Applicant also added new claim 26 which recites that the user-specified parameter is "voluntary". Support for this added claim is found, for example, at page 20, lines 16-17 of the application.

Applicant's amended independent claim 1 recites "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, a currently running context, corresponding to a first thread, in a specified microengine to let another context, corresponding to a different thread that is ready to execute, execute in that microengine and cause a different context and associated program counter to be selected, with the swapped first thread automatically re-enabled to run at some subsequent context arbitration point, and wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event." Thus, applicant context-swap instruction is a user instruction in which a user specifies the parameter

that is used to determine which event will wake-up a swapped-out thread to cause that thread to resume execution.

In contrast, none of the references cited by the examiner discloses at least the feature of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event."

Specifically, Moller discloses a SWAP microinstruction (Moller's col. 30, line 40). Moller explains that:

> **FIG. 1 illustrates the typical microprocessor system architecture used in a programmable digital device and can be divided into two distinct sections: a control and instruction acquisition and processing section A on the left and a data acquisition and manipulation section B on the right. Section A has, as its heart, the microprogram sequence controller of the present invention, generally designated with the reference numeral 10. The data acquisition and manipulation section B includes the data processing circuits, generally designated as 12, which includes the working registers 14, the arithmetic logic unit (ALU) 16 and the status register 18. The data processing circuits 12 process data acquired from memory 20 by performing whatever operations are required by the "machine" instruction pointed to by the data acquisition circuits, generally designated as 21, as addressed by program counter 22 and accessed via the memory address register 24. Each "machine" instruction is implemented on the microprocessor by a sequence of microinstructions selected by the microprogram sequence controller 10.**
>
> **While, for the purposes of discussion, certain of the blocks shown in FIG. 1 are shown explicitly interconnected by designated lines, it is to be understood that data communication between any of the blocks shown can be effected along the bidirectional data bus 26. Similarly addresses can be communicated between any of the blocks shown, on address bus 28.**
>
> **Microprogram sequence controller 10 generates addresses that determine the sequence of microinstructions that ultimately issue from the microprogram memory 30. The addresses generated by the microprogram sequence controller 10 are conducted from DATA.sub.-- OUT terminals of controller 10 on a "Y" address bus 32 to the address circuits (not shown) of the microprogram memory 30. With the application of such microinstruction addresses, the microprogram memory 30 will issue, on the microinstruction bus 34, microinstructions, usually of a word length on the order of 32 or more bits. (emphasis added, col. 4, line 33, to col. 5, line 3)**

Thus, a microinstruction is but a single step in the execution of one (1) machine instruction, and it configures the microprocessor in a particular way (e.g., activating a bus or some control signal). Moller's microinstructions, including the SWAP microinstruction, are not user instructions, or any type of instruction, and Moller's microinstructions do not include user-specified parameters. Indeed, microinstructions, such as those described in Moller, do not include parameters since those microinstructions are hard-wired control sequences designed to electrically connect the processor in some specific pre-determined manner. For example, Moller's SWAP microinstruction causes a CMUXCTL signal to be generated to cause multiplexer C-MUX 118 to pass the top of the LIFO stack 136 (LIFO stack 136 facilitates interrupt handling functions) to a down counter 120 (col. 30, lines 39-44). No parameters are used in conjunction with the SWAP microinstruction. Moller, therefore, fails to disclose or suggest at least the feature of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," as required by applicant's independent claim 1.

Bitar describes a system and method for context switching between a first and a second execution entity (such as a thread) without having to enter into protected kernel mode (Abstract). Bitar also describes an exemplary procedure for effecting a context switch between two threads (see FIG. 9, and accompanying description in col. 13, line 35, to col. 14, line 64), part of which may be implemented using a sequence of assembly instructions. But Bitar does not describe a single context-swap instruction that causes a context switch, nor does it describe a user-specified parameter, indicative of an occurrence of an event used to wake up a swapped-out context, for use with a context-swap instruction. Accordingly, Bitar does not disclose or suggest at least the features of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," as required by applicant's independent claim 1.

Adkins describes a scheduler executing on a serial communications adapter that schedules tasks at different priority levels (Abstract). Adkins describes that a sleeping task can be awakened and have its context restored when a new event associated with that task arrives at the port response queue (col. 8, lines 36-57). However, Adkins does not describe use of program instructions to control swapping, nor does not describe user-specified parameters indicative of an occurrence of an event used to wake-up a swapped out thread. Accordingly, Adkins does not disclose or suggest at least the features of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," as required by applicant's independent claim 1.

Colleran describes a method and system for foreground window determination of windows displaying applications in a displayed desktop (Abstract). While Colleran describes threads, Colleran does not describe instructions, let alone a context-swap instruction, to effect context switching. Colleran also does not describe use of a user-specified parameter, indicative of an occurrence of an event to wake-up a swapped-out thread, that is used in conjunction with a context swap instruction. Accordingly, Colleran does not disclose or suggest at least the features of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," as required by applicant's independent claim 1.

Because none of the references cited by the examiner discloses or suggests, alone or in combination, at least the features of "directing the processor having a plurality of microengines to swap, based on a user-specified parameter specified in a context-swap instruction, … wherein directing the processor comprises waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," applicant's independent claim 1 is patentable over the cited art.

Claims 4-8 and 10-14 depend from independent claim 1 and are therefore patentable for at least the same reasons as independent claim 1.

Independent claims 15, 22 and 24 recite "receiving a user-specified parameter specified in a context-swap instruction; performing a swapping operation to cause an executing context process corresponding to a first thread to be swapped with a different context and associated program counter, corresponding to a different thread that is ready to execute, the swapped first thread being automatically re-enabled to run at some subsequent context arbitration point; and waking up the swapped out context when the user-specified parameter specified in the context-swap instruction is activated, with the user-specified parameter specifying an occurrence of an event," or similar language. For reasons similar to those provided with respect to independent claim 1, at least these features are not disclosed by the cited art. Applicant's independent claims 15, 22 and 24 are therefore patentable over the cited art.

Claims 16 and 18-21 depend from independent claim 15 and are therefore patentable for at least the same reasons as independent claim 15.

It is believed that all the rejections and/or objections raised by the examiner have been addressed.

In view of the foregoing, applicant respectfully submits that the application is in condition for allowance and such action is respectfully requested at the examiner's earliest convenience.

All of the dependent claims are patentable for at least the reasons for which the claims on which they depend are patentable.

Canceled claims, if any, have been canceled without prejudice or disclaimer.
Any circumstance in which the applicant has (a) addressed certain comments of the examiner does not mean that the applicant concedes other comments of the examiner, (b) made arguments for the patentability of some claims does not mean that there are not other good reasons for patentability of those claims and other claims, or (c) amended or canceled a claim does not mean that the applicant concedes any of the examiner's positions with respect to that claim or other claims.

Enclosed is a Petition for One Month Extension of Time. The fees in the amount of $120 are being paid concurrently on the Electronic Filing System (EFS) by way of Deposit Account authorization. Please apply any other required fees to deposit account 06-1050, referencing the attorney docket number shown above.

Respectfully submitted,

Date: Feb. 9, 2007

Ido Rabinovitch
Attorney for Intel Corporation
Reg. No. L0080

Customer No. 20985
Fish & Richardson P.C.
Telephone: (617) 542-5070
Facsimile: (617) 542-8906

21555286.doc